10-708: Probabilistic Graphical Models, Spring 2015

12 : Variational Inference I

Lecturer: Eric P. Xing *Scribes: Fattaneh Jabbari, Eric Lei, Evan Shapiro*

1 Introduction

Probabilistic inference is one of the main tasks in graphical modeling, which is concerned with answering queries like marginal and conditional probabilities of the model. Given a distribution P defined by a graphical model over a set of variables V, we are interested in solving the following inference problems:

- Likelihood of observed data
- Marginal distribution $P(x_A)$ of a particular subset of nodes $A \subset V$
- Conditional distribution $P(x_A|x_B)$ of disjoint subsets $A, B \subset V$
- Mode of the density $\hat{x} = \arg\max_{x \in \chi^m} P(x)$

So far, we have seen several exact inference algorithms such as brute force enumeration, variable elimination, message passing (sum-product, belief propagation), and junction tree algorithms. Brute force approach applies summation over all variables except the query ones, while variable elimination applies a systematic ordering on summation operations by exploiting the graph structure. Both methods, however, are wasteful since each individual query computations are treated independently, and consequently, they neglect to share intermediate terms. One efficient alternative is message passing algorithm that shares intermediate terms and solves all local inference problems. All mentioned algorithms work only for tree structures and might not converge for loopy graphs, or even if they converge, we are not sure if the solution makes sense. Junction tree algorithm provides a way to convert any arbitrary loopy graph to a clique trees and then finds the exact solution to the inference problem by running a message passing on clique tree. While junction tree satisfies local and global consistency, it is expensive and exponential to the number of nodes in the largest clique. In this lecture, two main types of variational approximate inference techniques are introduced: loopy belief propagation and mean field approximation.

2 A Review on Belief Propagation

The whole story begins with a very intuitive heuristic again: message passing and belief propagation. At each iteration, each node i passes a message to each of its neighbors j when it receives all messages from its own neighbors (Figure 1a). This is called message update rule and can be calculated according to the following equation:

$$m_{i \to j}(x_j) \propto \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_i(x_i) \prod_{k \in N(i) \setminus j} m_{k \to i}(x_i)$$

where $\psi_{ij}(x_i, x_j)$ is called compatibilities or doubleton, $\psi_i(x_i)$ is called external evidence or singleton, and $m_{k \to i}(x_i)$ is message from all other neighbors of i except j.

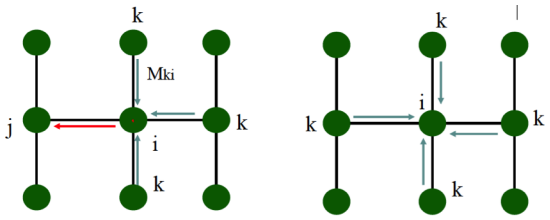


Figure 1: Belief propagation: (a) message passing, (b) node marginal

After the convergence, we can compute the marginal probabilities of every node using the following formula (Figure 1b), which are proved to be exact on tree structures.

$$b_i(x_i) \propto \psi_i(x_i) \prod_{k \in N(i)} m_k(x_k)$$

We can make message passing even more general by applying it to a factor graph. Therefore, the algorithm involves two types of messages: from node i to factor node a, and from factor a to node i (Figure 2):

$$m_{i \to a}(x_i) = \prod_{c \in N(i) \setminus a} m_{c \to i}(x_i)$$

$$m_{a \to i}(x_i) = \sum_{X_a \setminus x_i} f_a(X_a) \prod_{j \in N(a) \setminus i} m_{j \to a}(x_j)$$

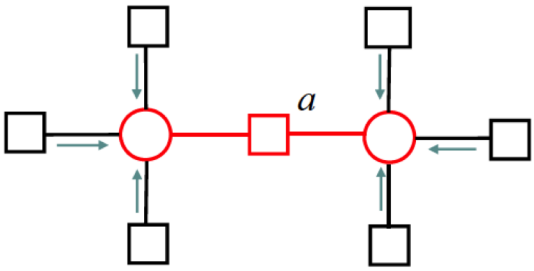


Figure 2: Belief propagation in factor graph

3 Loopy Belief Propagation

NOw, the question is what if we have an arbitrary graph? How do we do inference while we know junction tree is expensive? A clever engineering way is to run the same forward/backward message passing algorithm on every edge regardless of the graph type (tree or not tree). This is called loopy belief propagation, since the message circulate, and is in fact a fixed point iterative procedure that tries to minimize F_bethe (introduced later). More specifically, the following steps are repeated until convergence:

$$b_i(x_i) \propto \prod_{a \in N(i)} m_{a \to i}(x_i)$$

$$b_a(X_a) \propto f_a(X_a) \prod_{i \in N(a)} m_{i \to a}(x_i)$$

$$m_{i \to a}(x_i) = \prod_{c \in N(i) \setminus a} m_{c \to i}(x_i)$$

$$m_{a \to i}(x_i) = \sum_{X_a \setminus x_i} f_a(X_a) \prod_{j \in N(a) \setminus i} m_{j \to a}(x_j)$$

However, it is clear that it might not converge or may converge to a wrong solution. This was one the popular algorithms in late 90's and a substantial amount of research and experiments has been done to understand its theoretical behavior. Murphy et. al (1999), as a good example, have conducted a systematic case study and interestingly observed that a good approximation is still achievable empirically if:

- Stop after a fixed number of iterations
- Stop when no significant change in beliefs
- If solution is not oscillatory but converges, it usually is a good approximation

In the following section, we will discuss in more detail the theoretical properties and characteristics of loopy belief propagation.

4 Bethe Approximation to Gibbs Free Energy

4.1 Approximating a Probability Distribution

Observing the nice performance of belief propagation on inference in graphs with loops, we question whether this algorithm has any theoretical justification or whether it is a dirty hack. It will be shown that belief propagation finds an approximation to the true distribution that is roughly optimal. Now we lay the groundwork for finding an optimal approximation to a distribution. Formally, let P be the actual distribution over an arbitrary graph,

$$P(X) = \frac{1}{Z} \prod_{f_a \in F} f_a(X_a).$$

We wish to find a distribution Q that approximates P. Our definition of an optimal approximation is one that minimizes KL-divergence,

$$KL(Q_1||Q_2) = \sum_X Q_1(X) \log \left(\frac{Q_1(x)}{Q_2(x)} \right)$$

which can be interpreted as a distance between distributions. Not a true metric, it satisfies

$$KL(Q_1||Q_2) \geq 0$$

$$KL(Q_1||Q_2) = 0 \iff Q_1 = Q_2$$

but not symmetry. Thus we can choose between $KL(P||Q)$ and $KL(Q||P)$. Unfortunately $KL(P||Q)$ requires inference on P. On the other hand, without inference we can write

$$KL(Q||P) = \sum_X Q(X) \log \left(\frac{Q(X)}{P(X)} \right)$$

$$= \sum_X Q(X) \log Q(X) - \sum_X Q(X) \log P(X) = -H_Q(X) - E_Q \log P(X)$$

$$= -H_Q(X) - E_Q \log \left(\frac{1}{Z} \prod_{f_a \in F} f_a(X_a) \right) = -H_Q(X) - \sum_{f_a \in F} E_Q \log f_a(X_a) + \log Z$$

where H_R denotes entropy of distribution R.

We denote the first two terms by the *free energy* $F(P, Q)$,

$$F(P, Q) = -H_Q(X) - \sum_{f_a \in F} E_Q \log f_a(X_a).$$

By inspection, $F(P, Q) \geq F(P, P) = -\log Z$. Let us consider how to compute $F(P, Q)$. The entropy H_Q is intractable in general because we must sum over all possible values of X. In contrast, $E_Q \log f_a(X_a)$ is computable if the marginal of f_a is known. Altogether, F is hard to compute. Therefore, our new approach is to approximate F by some easily computable $\hat{F}$. We shall see that a choice approximation turns out to be the Bethe approximation to Gibbs free energy.

4.2 Deriving the Bethe Approximation

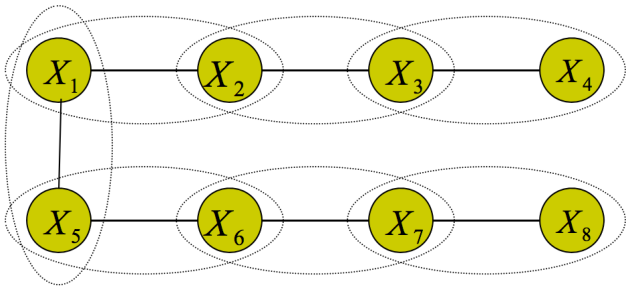


Figure 3: A line graph, the simplest kind of tree.

To motivate the Bethe approximation, consider the tree-structured distribution in Fig. 3. The joint probability is

$$P(X_1, \ldots, X_8) = P(X_8|X_1, \ldots, X_7)P(X_1, \ldots, X_7) = P(X_8|X_7)P(X_7|X_1, \ldots, X_6)$$

$$= \cdots = P(X_8|X_7)P(X_7|X_6)P(X_6|X_5)P(X_5|X_1)P(X_1|X_2)P(X_2|X_3)P(X_3, X_4)$$

$$= \frac{P(X_8, X_7)P(X_7, X_6)P(X_6, X_5)P(X_5, X_1)P(X_1, X_2)P(X_2, X_3)P(X_3, X_4)}{P(X_7)P(X_6)P(X_5)P(X_1)P(X_2)P(X_3)}.$$

Extrapolating this structure, we see that for any tree the distribution is

$$b(\mathbf{x}) = \prod_a b_a(\mathbf{x}_a) \prod_i b_i(x_i)^{1-d_i}$$

where a enumerates edges, i enumerates vertices, and d_i is the degree of vertex i. We can now compute the entropy as well as the free energy with an arbitrary distribution $P(X) = \frac{1}{Z} \prod_{f_a \in F} f_a(X_a)$. We have

$$H_{tree} = -\sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \log b_a(\mathbf{x}_a) + \sum_i (d_i - 1) \sum_{x_i} b_i(x_i) \log b_i(x_i)$$

$$F_{tree} = \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \log \frac{b_a(\mathbf{x}_a)}{f_a(\mathbf{x}_a)} + \sum_i (1 - d_i) \sum_{x_i} b_i(x_i) \log b_i(x_i).$$

The free energy only requires summation over edges and vertices, so it is easy to compute. For an arbitrary graph, however, the free energy is far harder to write down. Therefore, we use the Bethe approximation, which has the exact same formula as free energy for a tree,

$$F_{Bethe} = \sum_a \sum_{\mathbf{x}_a} b_a(\mathbf{x}_a) \log \frac{b_a(\mathbf{x}_a)}{f_a(\mathbf{x}_a)} + \sum_i (1 - d_i) \sum_{x_i} b_i(x_i) \log b_i(x_i).$$

The advantage of the Bethe approximation, as mentioned previously, is that it is easy to compute for any graph. The disadvantage, however, is that there is an ambiguous relationship to the true free energy. In general, $\hat{F}(P, Q)$ could be greater, equal, or less than $F(P, Q)$. An intuitive argument can be made that the closer the underlying graph is to a tree, the more accurate the Bethe approximation. However, there are no rigorous, general results.

To recap, we aim to minimize KL-divergence between P and Q. We can approximate this quantity by the Bethe free energy. The next step is to find the $\{b_a\}$ and $\{b_i\}$ that attain the minimum such that local consistency holds:

$$\forall i, x_i \quad \sum_{x_i} b_i(x_i) = 1$$

$$\forall a, i \in N(a), x_i \quad \sum_{\mathbf{x}_a | \mathbf{x}_a = \{x_i, x_j\}} b_a(\mathbf{x}_a) = b_i(x_i)$$

where $N(a) = \{i : x_i \in \mathbf{x}_a\}$. By the method of Lagrange multipliers, the objective function becomes

$$L = F_{Bethe} + \sum_i \gamma_i \left(1 - \sum_{x_i} b_i(x_i)\right)$$

$$+ \sum_a \sum_{i \in N(a)} \sum_{x_i} \lambda_{ai}(x_i) \left(b_i(x_i) - \sum_{\mathbf{x}_a | \mathbf{x}_a = \{x_i, x_j\}} b_a(\mathbf{x}_a)\right).$$

Assuming that F_{Bethe} is convex, we now find the stationary points:

$$0 = \frac{\partial L}{\partial b_i(x_i)} = 1 + \log b_a(\mathbf{x}_a) - \log f_a(\mathbf{x}_a) - \sum_{a \in N(i)} \lambda_{ai}(x_i)$$

$$\implies b_i(x_i) = \exp\left(\frac{1}{d_i - 1} \sum_{a \in N(i)} \lambda_{ai}(x_i) - \frac{\gamma_i}{d_i - 1} - 1\right)$$

$$\propto \exp\left(\frac{1}{d_i - 1} \sum_{a \in N(i)} \lambda_{ai}(x_i)\right)$$

$$0 = \frac{\partial L}{\partial b_a(X_a)} = 1 + \log b_a(\mathbf{x}_a) - \log f_a(\mathbf{x}_a) - \sum_{i \in N(a)} \lambda_{ai}(x_i)$$

$$\implies b_a(\mathbf{x}_a) = \exp\left(\log f_a(\mathbf{x}_a) + \sum_{i \in N(a)} \lambda_{ai}(x_i) - 1\right)$$

$$\propto \exp\left(\log f_a(\mathbf{x}_a) + \sum_{i \in N(a)} \lambda_{ai}(x_i)\right).$$

5 Belief Propagation as Minimizing Bethe Free Energy

From the Bethe Free Energy we can obtain the Belief Propagation equations for a factor graph. For the variables of the factor graph, we have:

$$b_i(x_i) \propto exp\left(\frac{1}{d_i - 1}\sum_{a \in N(i)} \lambda_{ai}(x_i)\right)$$

With $\lambda_{ai} = log(m_{i\to a}(x_i)) = log\left(\prod_{a \in N(i) \neq a} m_{b\to i}(x_i)\right)$, this gives us, for the variables of the factor graph,

$$b_i(x_i) \propto f_i(x_i) \prod_{a \in N(i)} m_{a\to i}(x_i)$$

For the factors, we have:

$$b_a(X_a) \propto exp\left(-log(f_a(X_a)) + \sum_{i \in N(a)} \lambda_{ai}(x_i)\right)$$

Using our λ_{ai}, for the factors it gives us:

$$b_a(X_a) \propto f_a(X_a) \prod_{i \in N(a)} \prod_{c \in N(i) \setminus a} m_{c\to i}(x_i)$$

Now we define $b_{a\to i}(x_i) = \sum_{X_a \setminus x_i} b_a(X_a)$. Using the b_a derived above,

$$m_{a\to i}(x_i) = \sum_{X_a \setminus x_i} f_a(X_a) \prod_{j \in N(a) \setminus i} \prod_{b \in N(j) \setminus a} m_{b\to j}(x_j)$$

6 Loopy Belief Propagation

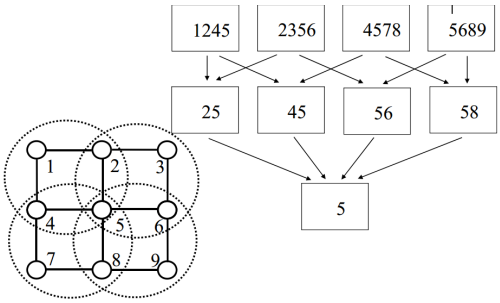
Finding the true distribution is difficult. So instead we'll optimize over an easier distribution q. We will find

$$q* = argmin_{q \in S}(F_{betha}(p, q))$$

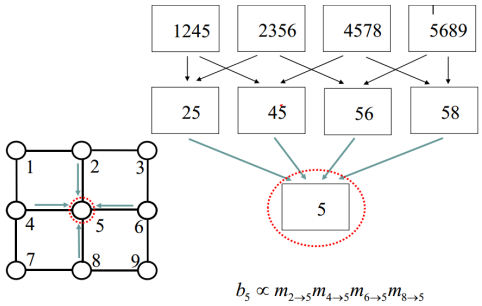
Now we can just optimize H_q. However, optimizing H_q is still difficult. Instead we will optimize the bethe free energy of the beliefs $F(b)$. Loopy belief propagation can be interpreted as a method that uses fixed point iteration to optimize $F(b)$. Note that we do not know the precise relationship between optimizing $F(b)$ and optimizing H_q. Oftentimes loopy belief propagation does not converge to the correct solution, although there are a wide class of problems for which it frequently converges to the correct distribution

7 Generalized Belief Propagation

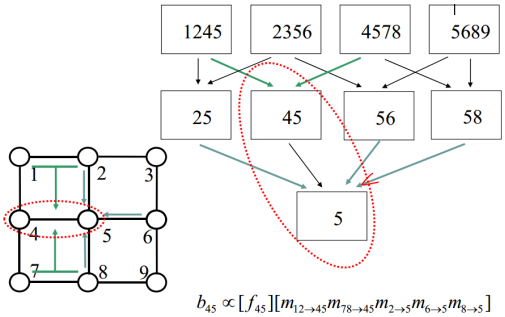
Using our insights from Bethe free energy, we can choose more accurate approximations of H_q and achieve better results. We will consider regions of the graph. In normal loopy belief propagation, regions of the graph are single nodes. In general belief propagation, beliefs are calculated over multiple regions and then integrated. Instead of using Bethe free energy, this more general method uses Gibbs free energy. This region based approach was found by Kikuchi, and provides better approximations of the true free energy than Bethe free energy, at increased computational cost. We can see how this region integration is performed by understanding the following example: In this example we have four separate regions of the graph. Each region is comprised of four nodes.

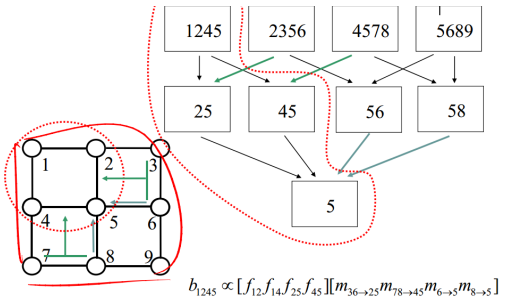


Beliefs are calculated and integrated hierarchically. This means there is increased computational cost.



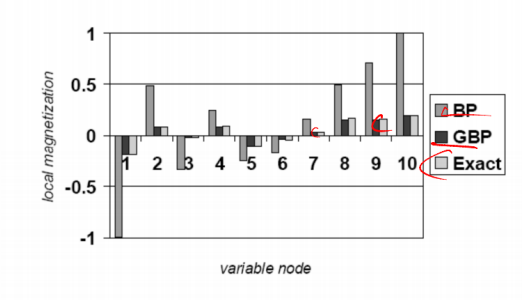
Looking at a particular region, we see how nodes that share regions have their energies calculated from their parent regions. By increasing the number of regions, we increase the accuracy of the approximation. However, that creates more regions and causes more nodes to be members of multiple regions.





On many problems using GBP instead of standard loopy belief propagation can lead to far better results, especially if loopy belief propagation does not converge. This follows theoretically, but it is nice to see

empirically that the improvement is significant.



@inproceedingsmurphy1999loopy, title=Loopy belief propagation for approximate inference: An empirical study, author=Murphy, Kevin P and Weiss, Yair and Jordan, Michael I, booktitle=Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, pages=467–475, year=1999, organization=Morgan Kaufmann Publishers Inc.